CMSC 151: Introduction to Computer Science I

The University of Chicago, Fall 2013, John Reppy and Adam Shaw

`http://www.classes.cs.uchicago.edu/archive/2013/fall/15100-1`

Welcome! In CS151, we introduce a selection of major computer science topics through instruction in computer programming and various analytical techniques.

CS151 is designed for students intending to major or minor in the subject, although others are welcome.

The specific goals of the course are these:

- to understand solving computational problems in terms of identifying, and, when necessary, designing, relevant abstractions,
- to process data structures in several ways, most importantly by the technique of structural recursion,
- to learn to recognize and exploit common computational patterns through code organization and higher-order programming,
- to learn to use simple and polymorphic types as a powerful approximation of correctness in computer programs, and
- to analyze the efficiency of certain algorithms.

In pursuing these goals, students will become acquainted with a selection of classic data structures and algorithms. Broader, more technical treatments of these topics, in particular algorithm analysis, are presented in later undergraduate courses.

We use the Racket programming language in our studies. Racket is a dialect of Scheme, a language with a long history in the field of computer science generally and college-level instruction specifically.

Having completed this course, students will know how to use computer programming as an elegant, robust and efficient method for analytical problem solving and creative endeavors. Furthermore, students will have begun to develop an understanding of the programming's role in the larger context of computer science. Students will discover, in future work, that the experience gained in this course applies to programming generally, in any programming language; that is, CS151 should not be thought of as a course in programming Scheme. Furthermore, students will have gained a experience with some known best practices in the discipline.

Things You Must Do More or Less Immediately

- **Request a CS account no later than Monday, September 30.** Having a CS account allows you to use CS department machines, supplies you a home directory securely accessible from anywhere on campus, and indeed from anywhere on the Internet, and various other perquisites. You request an account by filling out the web form at the following location:

  `https://www.cs.uchicago.edu/info/services/account_request`

- Register with *piazza*. Piazza is an online question-and-answer system that we use for that purpose as well as distribution of course materials on occasion. You will receive an email about piazza registration, with instructions, at your `uchicago` email address at the start of the quarter, so make sure you check that email address by September 30.

Instructors

John Reppy, email: `jhr@cs.uchicago.edu`, office: Ryerson 256.

Adam Shaw, email: `ams@cs.uchicago.edu`, office: Ryerson 157.

Graduate Teaching Assistants

Erik Bodszar, Charisee Chiw, Nicholas Seltzer, Nedelina Teneva, Fan Yang, Liwen Zhang, Zhixuan Zhou.

Contacting Us

If you have questions about the course, and those questions are in a sense impersonal — that is, they are about course material or course logistics — we ask that you post those questions publicly on piazza, rather than contacting any of the staff members directly. This ensures you will receive the fastest, most consistent possible response from the staff. Since students usually have *common* questions, posting public questions is also very efficient for your classmates as well. As yet another advantage, it avoids duplication of work on the part of the staff.

In cases where you have a question that is about your own personal situation and not relevant to the class as a whole, you may ask a "private question" on piazza, which is invisible to your classmates, or send email to your instructor directly.

Lectures All lectures are in Ryerson 251. There are two sections.

- Section 1: MWF 10:30–11:20, John Reppy.
- Section 2: MWF 11:30–12:20, Adam Shaw.

The first lecture is on Monday, September 30; the last is on Wednesday, December 4.

We do not allow the use of electronic devices during lectures. The devices are simply too distracting. This includes laptops, smartphones, and tablets. The lone exception to this policy is for students whose handwriting issues necessitate their use of a device for note taking, who will be permitted to use a plain text editor on a laptop whose wireless capability is turned off. If you are such a student, let your instructor know.

Lab Sessions Students must register for and attend lab sessions each week. Lab sessions are held in the Computer Science Instructional Laboratory (also known as the CSIL). There is a new CSIL facility as of this quarter; it is located on the first floor of Crerar Library. Attendance at the lab session for which you are registered is mandatory.

We offer twelve weekly lab sections at eight different meeting times. During the four Wednesday slots, two labs occur side by side in adjacent lab quadrants. If you need to switch your lab time, there will be a way to do so online early in the quarter; details to follow. You will work on a department's Macintosh computer during your lab session. You must use the department's computer during lab and may not use your own laptop.

The lab times are as follows:

> Tues 12pm–1:20pm; Tues 1:30pm–2:50pm; Tues 3pm–4:20pm; Tues 4:30pm–5:50pm; Wed 12:30pm–1:50pm; Wed 2pm–3:20pm; Wed 3:30pm–4:50pm; Wed 5pm–6:20pm.

There will be no lab exercises during the week of Thanksgiving, and no lab exercises the last week (before reading period).

Schedule of Topics by Week (subject to change)

Week	**Topics**
1	expressions, functions, types
2	structures, variants, lexical scope
3	linked lists, structural recursion
4	higher-order programming, parametric polymorphism
5	trees and tree algorithms
6	accumulators, mutual recursion
7	efficiency and efficiency analysis, sorting
8	maps, state, hash tables
9	state, graphs, graph algorithms
10	graph algorithms

Office Hours To be announced on the web once the quarter starts. In addition to the office hours we provide ourselves, the College Core Tutor Program employs computer science tutors Sunday through Thursday nights from 7pm–11pm, starting in the second week.

Text *How to Design Programs*, Felleisen *et al.*, ISBN 0-262-06218-6. The textbook is available on campus at the Seminary Co-op Bookstore[1]; you can of course find new and used copies at your favorite online bookstore as well. Before buying a copy, note that the full text of the book is available online at `http://www.htdp.org` free of charge.

Software All the software we use in this course is available free of charge for all common platforms. We will mainly use *DrRacket*, available at `http://racket-lang.org`, and *subversion*. Macintosh and Linux users very likely have subversion on their machines already. Windows users will need to download and install *Cygwin*, and will be able to include subversion in their Cygwin installations. We will provide detailed installation instructions for various platforms once the quarter starts.

Grading Coursework is comprised of lab exercises (done at lab sessions, discussed above), homework assignments, projects, and exams. The relative weighting of these in computing your grade is below.

Homework There will be weekly homework assignments. These will be assigned on Monday or Tuesday and will be due the following Monday.

Projects There will be a longer multipart project during the latter part of the term. The final part of this project will be due during exam week.

Exams There will be two exams for all students at the following dates and times: Wednesday, October 30, 7pm–9pm, and Wednesday, December 4, 7pm–9pm. Please plan accordingly. The location is to be announced. There will be no exam during finals week. Also, there will be no lab sessions during the weeks exams are given.

Each student's final grade will be computed according to the following formula: homework and project work 30%, labs 20%, exams 25% each. We will curve the grades, so what precisely constitutes an A, B, *etc.* will be determined by the collective performance of the class.

Late Work Deadlines in this course are rigid. Since you submit your work electronically, deadlines are enforced to the minute. Late work will not be counted, with the following exception. You have one 24-hour extension on any lab or homework assignment (except the first), no questions asked. Note the 24-hour extension may not be used on the first homework or lab exercise. We will let you know the details of how to request an extension during the quarter.

[1] 5751 S. Woodlawn Avenue; `http://www.semcoop.com`.

(We will also accept late work in the case of special circumstances, when those circumstances are extraordinary.)

Academic Honesty In this course, as in all your courses, you must adhere to college-wide honesty guidelines as set forth at `http://college.uchicago.edu/policies-regulations/academic-integrity-student-conduct`. The college's rules have the final say in all cases. Our own paraphrase is as follows:

1. Never copy work from any other source and submit it as your own.
2. Never allow your work to be copied.
3. Never submit work identical to another student's.
4. Document all collaboration.
5. Cite your sources.

We are serious about enforcing academic honesty. If you break any of these rules, you will face tough consequences. Please note that sharing your work publicly (such as posting it to the web) definitely breaks the second rule. With respect to the third rule, you may discuss the general strategy of how to solve a particular problem with another student (in which case, you must document it per the fourth rule), but you may not share your work directly, and when it comes time to sit down and start typing, you must do the work by yourself. If you ever have any questions or concerns about honesty issues, raise them with your instructor, early.

Advice Writing code that does what it is supposed to do can be joyful, even exhilarating. By contrast, fighting for hours with broken code is misery. We would like you to help you experience more of the former and less of the latter. Work methodically. Start your work well ahead of time. Beyond a certain point, it is not profitable to be stumped. If you have made no progress in some nontrivial chunk of time, say, one hour, it is time to stop and change your approach. Use one of our many support mechanisms to get some assistance. We will help you get going again when you are stuck.

2013 September 29 5:30pm. This is revision 3 of this document.